Advances in Computational Design, *Vol. 1, No. 2 (2016) 175-194*
DOI: http://dx.doi.org/10.12989/acd.2016.1.2.175

Dolphin Echolocation Optimization: Continuous search space

A. Kaveh* and N. Farhoudi

Centre of Excellence for Fundamental Studies in Structural Engineering, School of Civil Engineering, Iran University of Science and Technology, Tehan-16, Iran

(Received October 29, 2015, Revised December 7, 2015, Accepted December 8, 2015)

Abstract. Nature has provided inspiration for most of the man-made technologies. Scientists believe that dolphins are the second to humans in smartness and intelligence. Echolocation is the biological sonar used by dolphins for navigation and hunting in various environments. This ability of dolphins is mimicked in this paper to develop a new optimization method. Dolphin Echolocation Optimization (DEO) is an optimization method based on dolphin's approach for hunting food and exploration of environment. DEO has already been developed for discrete optimization search space and here it is extended to continuous search space. DEO has simple rules and is adjustable for predetermined computational cost. DEO provides the optimum results and leads to alternative optimality curves suitable for the problem. This algorithm has a few parameters and it is applicable to a wide range of problems like other metaheuristic algorithms. In the present work, the efficiency of this approach is demonstrated using standard benchmark problems.

Keywords: Dolphin Echolocation Optimization; continuous search space; mathematical examples; truss structure

1. Introduction

Optimization is selection of the best element from some set of available alternatives. Mathematical programming and metaheuristic algorithms are two main approaches for optimization. Mathematical programming includes a wide range of methods such as linear programming, Nonlinear Programming, Stochastic Programming and Dynamic Programming. These methods perform local search with higher accuracy in comparison to stochastic methods; however they require gradient information and an initial starting point to perform properly. Moreover variables and fitness function should be continuous.

Metaheuristic optimization methods are the recent generation of optimization methods. These methods are inspired from natural phenomena and their capabilities in optimal design of structures are demonstrated in many research studies. Evolutionary algorithm (EA) proposed by Fogel *et al*. (1966), Koza (1990) and Genetic algorithm (GA) proposed by Holland (1975) and Goldberg (1989) is inspired by Darwin's theory of evolution. Particle Swarm Optimization (PSO) proposed by Eberhart and Kennedy (1995) mimics the social interaction behavior of birds flocking and fish schooling. Ant colony optimization (ACO) proposed by Dorigo *et al*. (1996) simulates the

*Corresponding author, Professor, E-mail: alikaveh@iust.ac.ir

 ISSN: 2383-8477 (Print), 2466-0523 (Online)

foraging behavior of real life ant colonies that can establish a shortest route from food source to their nest and vice versa. Harmony Search (HS) method invented by Li and Geem (2004) imitates the musical performance process which takes place when a musician searches for a better state of harmony. Big Bang-Big Crunch algorithm (BB-BC) proposed by Erol and Eskin (2006) relies on the theory of evolution of the universe, Cuckoo Search (CS) by Yang and Deb (2009) is based on the obligate brood parasitic behavior of some Cuckoo species, Charged System Search (CSS) proposed by Kaveh and Talatahari (2010) utilizes the Newtonian and electrical physics laws. Ray Optimization (RO) was developed by Kaveh and Khayatazad (2013) that utilizes the refraction of light described by Snell's law. Colliding Bodies Optimization (CBO) was developed by Kaveh and Mahdavai (2014) which is based on the governing laws of one dimensional collision between two bodies from the physics that one object collides with other object and they move toward minimum energy level. Bat-inspired algorithm presented by Yang (2010, 2011), Teaching-Learning-based optimization (TLBO) by Rao *et al*. (2011), Sadollah *et al*. (2015) developed Water Cycle, Mine Blast and improved mine blast algorithms, Gonçalves *et al*. (2015) presented Search Group Algorithm, and Mirjalili developed the Ant Lion Optimizer (2015). are other metaheuristic algorithms which have sources in nature. Some other applications of these algorithms may be found in (Kaveh and Zolghadr 2014, Kaveh and Maniat 2015, Kaveh and Ilchi Ghazaan 2015).

Dolphin Echolocation Optimization is one of the most recent metaheuristic algorithms. This method imitates dolphin's strategies for exploring their environment. DEO in discrete search space is developed in the work of Kaveh and Farhoudi (2013). In this study, DEO is extended for continuous search space.

Section 2 presents the real dolphins tactics in searching their environment. Section 3 introduces Dolphin Echolocation Optimization for continuous search space. Sections 4 and 5 are devoted to the application of DEO in solving some benchmark mathematical and mechanical examples. In Section 6 concluding remarks are provided.

2. Dolphin echolocation in nature

As early as 1947, Arthur McBride observed attempts to capture dolphins at night in turbid waters. He noted that the animals could avoid fine mesh nets and, even at distances beyond visual range, could detect openings in the nets. However, it was not until 1960 that Kenneth Norris and his colleagues unequivocally demonstrated echolocation in dolphins by covering a dolphin's eyes with rubber suction cups and observing that the animal could avoid obstacles in a maze-all the while emitting ultrasonic sounds. Nowadays, acousticians understand that dolphins and bats possess a sophisticated biosonar system that allows them not only to detect, discriminate, and pursue prey, but also to track the trajectory of prey in order to solve the prey-intercept problem. And those feats are typically accomplished in noisy environments often cluttered with background targets. In addition to catching prey, dolphins and bats also use biosonar to navigate and avoid obstacles (Au 2007).

A dolphin is able to generate sounds in the form of clicks. Frequency of these clicks is higher than that of the sounds used for communication and differs between species. When the sound strikes an object, some of the energy of the sound-wave is reflected back towards the dolphin. As soon as an echo is received, the dolphin generates another click. This process is depicted in Fig. 1. The time lapse between click and echo enables the dolphin to evaluate the distance from the object; the varying strength of the signal as it is received on the two sides of the dolphin's head

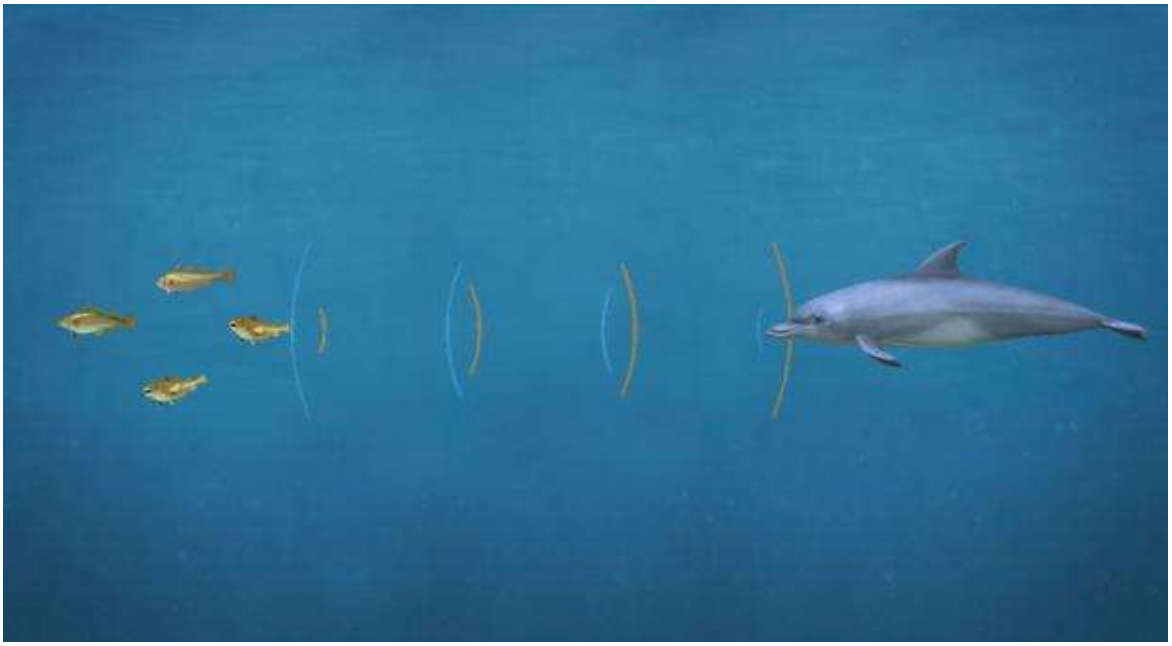

Fig. 1 A real dolphin catching its prey

makes it possible for him to evaluate the direction. By continuously emitting clicks and receiving echoes in this way, the dolphin can track objects and home in on them, May (1990). The clicks are directional and are for echolocation, often occurring in a short series called a click train. The click rate increases when approaching an object of interest, Au (2007).

3. Dolphin Echolocation Optimization in continuous search space

Dolphin Echolocation Optimization is recently developed for the discrete search space by the authors (Kaveh and Farhoudi 2013). Dolphins take advantages of echolocation to discover their environment. The problem of finding some variables' value in a search space is like search of dolphins in their environment. In optimization choosing the best answer for a problem is similar to dolphin's attempt to find the best target. Dolphins at the outset, look around the search space to find out where the preys are, subsequently they restrict the trace in order to locate the precise position.

The method simulates dolphin echolocation by decreasing size of the random search space proportional to the distance to the target. In the proposed method, the user defines a curve on which the optimization convergence should be performed. In this way the convergence criteria is enforced to the algorithm and also this process makes the algorithm's convergence less parameter dependent.

There is a unified method for parameter selection in meta-heuristics in discrete search space (Kaveh and Farhoudi 2011). The method works by controlling an index of *convergence factor*. A Convergence Factor (CF) is defined as the average probability of the best answer. Here in continuous search space because of the inherently continuous characteristic of the variables it is

not possible to calculate probability for the best answer as a single point, instead standard deviation is chosen to be a criterion for convergence. For variable *j*, Convergence Factor (CF) is defined as follows

$$CF_j = 1 - \frac{SD_j}{(UL_j - LL_j)/2} \tag{1}$$

where, SD_j is the standard deviation of the j^{th} location values; UL_j is the upper limit of the j^{th} variable; and LL_j is the lower limit of the j^{th} variable.

3.1 Dolphin Echolocation Optimization algorithm

A curve according to which the convergence factor should change during the optimization process should be assigned. Here, the change of *CF* is considered to be according to the following curve

$$PP(Loop_i) = PP_1 + (1 - PP_1)\frac{Loop_i^{Power} - 1}{(Loops\ Number)^{Power} - 1} \tag{2}$$

PP: Predefined probability; PP_1: Predefined probability of the first loop. In other words, PP_1 is a guess for *PP* in first loop. It should be noted that, it would be better to let the algorithm to start its work with a random selection at first loop and calculate CF for this loop, however, as it is not of much importance, a value of around 10% may be proper for all cases, then generally there is no need to calculate it; $Loop_i$ Number of the loop in which optimization process is performing; *Power*: Degree of the curve. As it can be seen, the curve in Eq. (1) is of *Power* degree. Loops *Number*: Number of loops that algorithm should end its work and give the result. The loops number should be chosen by the user according to the computational effort that can be provided for the algorithm.

Fig. 2 shows the variation of *PP* by the changes of the *Power*, using the proposed formula, Eq. (2).

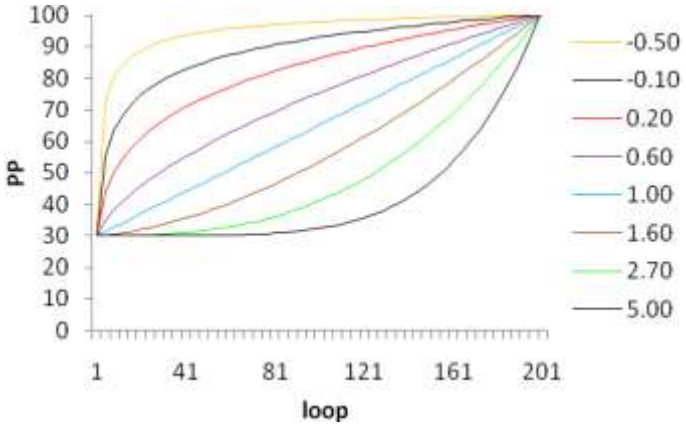

Fig. 2 Sample convergence curves, using Eq. (1) for different values for power (Kaveh and Farhoudi 2011)

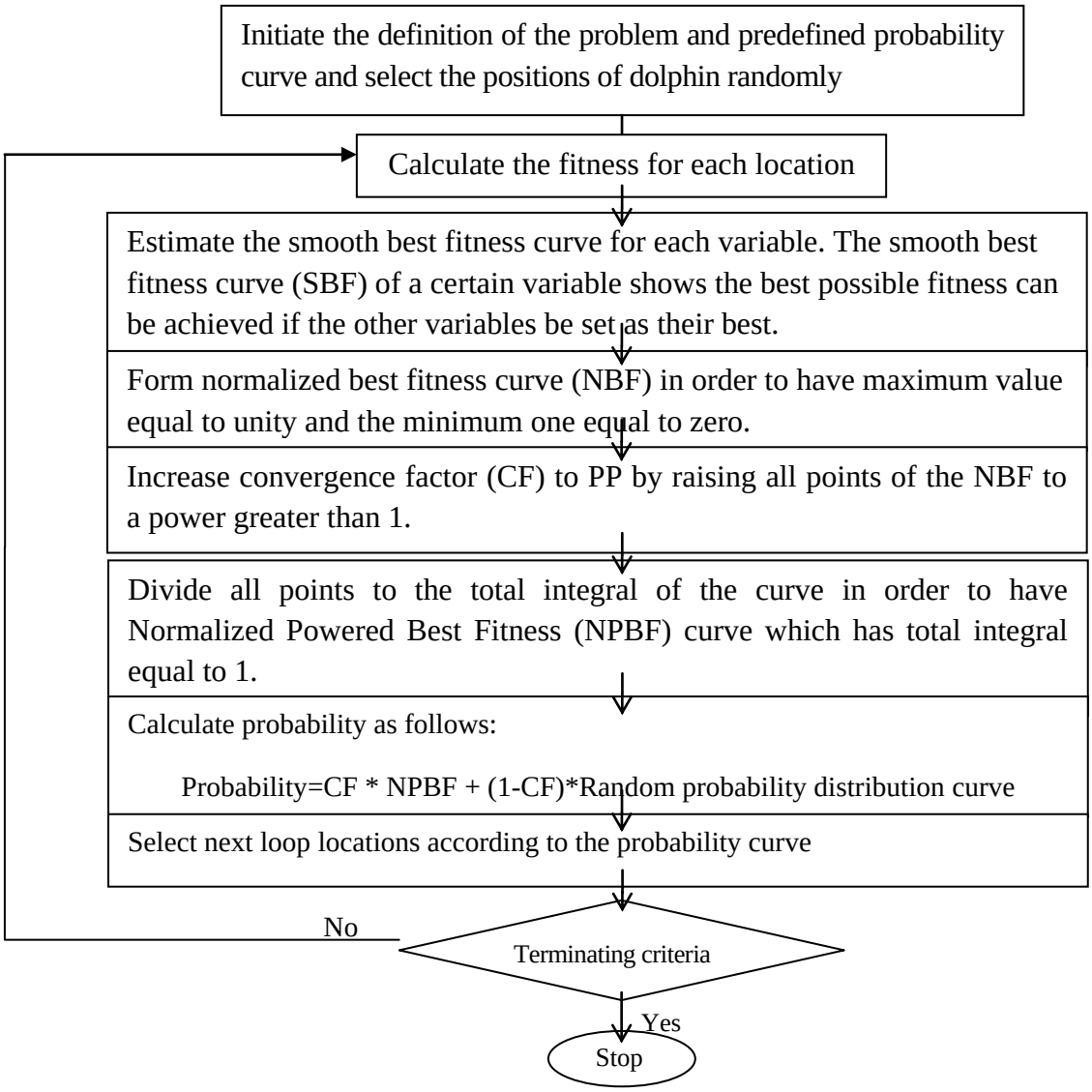

Fig. 3 The flowchart of the DEO algorithm

3.1.1 Mathematical formulation of the DEO algorithm

Steps of the DEO algorithm can be stated as follows:

1. Initiate *NL* locations for a dolphin randomly.
2. Calculate *PP* of the loop using Eq. (2).
3. Calculate the fitness of each location.

Fitness should be defined in a manner that better answers get higher values. In other words the optimization goal should be to maximize the fitness.

4. Create the best fitness matrix (BF), Leading curve (LC) and Smooth Best Fitness curve

(SBF) according to dolphin rules as follows:

4.1. Create the best fitness matrix (BF) and draw the leading curve (LC).

For j=1 to NV

For i=1 to NL

$$BF(L(i,j),j) = \max(Fitness(i), BF(L(i,j),j))$$

$$LC(\mathrm{L(i,j)}+x,j) = \begin{cases} \max(\frac{1}{R_e} * (R_e - |x|) * Fitness(i), LC(\mathrm{L(i,j)}+x,j)) & -\mathrm{R_e} \le x \le \mathrm{R_e} \\ 0 & otherwise \end{cases} \quad (3)$$

where, NV is number of variables; NL is number of locations; $L(i,j)$ is the j^{th} variable's value in the i^{th} location; BF contains the best ever achieved fitness for each variable. $LC(x,j)$ is the maximum value obtained by producing an inverse V-type curve on all locations of this loop by considering y-axis as fitness and x-axis as available values for the j^{th} variable. R_e is the effective radius which shows the distance around a selected alternative that its neighbors' probabilities are affected from its fitness. R_e is recommended to be not more than 1/4 of the search space; $Fitness(i)$ is the fitness of the i^{th} location.

3.2 Draw Smooth Best Fitness

Smooth Best Fitness (SBF) is a smooth curve for each variable which shows how each alternative is fitted for this variable. It passes through BF points that lay on /over LC curve. This can be done by different methods; however, the one used here follows the following steps to draw the curve for the j^{th} variable:

1. For the first alternative, value of SBF is the maximum value of LC and BF for this alternative. Or SBF_{1j}=max(LC_{1j}, BF_{1j});

Set alternative 1 as FiPo (First Point);

2. a) Form a group of points of maximum PoNum number X={X_1,X_2,..$X_{\mathrm{LastPoint}}$}| X_1>=FiPo+1 & $X_{\mathrm{LastPoint}}$<=UL_j (UL_j is the upper limit of the j^{th} variable) & LastPoint<=PoNumin which for each X_i $\in X$, $BF_{Xi,j}$ is greater or equal to $LC_{Xi,J}$.

* Here it should be noted that for the point like p, if $BF_{p,j}$ is still zero it means that alternative p has not been used for j^{th} variable so far, then p cannot be one of the X group members.

b) For each point of X group like X_i, calculate the slope of a line which connects FiPo to Xi.

c) Define the point for which the slope is maximized and name it MaxPo.

d) Draw a line from FiPo to MaxPo and set the value of $\mathrm{SBF_{FiPo+1,\,j}}$ to $\mathrm{SBF_{MaxPo,j}}$ to be on the line.

3. If MaxPo is not equal to UL_j set FiPo equal to FiPo+1 and repeat Steps 2 and 3.

5. Normalize the smooth best fitness curve in order to have maximum value equal to unity and the minimum one equal to zero. For the j^{th} variable, if maximum and minimum values of SBF be considered as Max_j and Min_j Normalized SBF or NBF will be equal to

$$\mathrm{NBF(x,\,j)} = (\mathrm{SBF(x,\,j)} - \mathrm{min_j})/(\mathrm{max_j} - \mathrm{min_j}) \quad (4)$$

where, x belongs to the j^{th} variable domain.

6. Normalized best fitness (NBF) curve is going to be used in this step as probability distribution curve but before that, its convergence factor should be changed according to the

predefined probability curve. An increase in convergence factor occurs when standard deviation decreases. Decreasing standard deviation is implemented in algorithm by raising all points of normalized best fitness curve to a power greater than 1. The power should changes till CF achieved from Eq. (1) be equal to predefined probability (PP).

7. After that, divide NBF to total area between the curve and *x* axis in order to have a curve with total area enclosed by the curve equal to unity. Name the curve as the Normalized Powered Best Fitness curve (NPBF).

8. Probability distribution curve is the CF * NPBF plus (1-CF)*Random distribution curve

(Random distribution curve is a curve with constant value of 1/(Domain length). It would be obvious that its integral all around the domain will be equal to 1).

9. Select locations of the next loop according to the probability curve. In order to performing selection, for each dimension of a specified location choose a random number between 0 and 1. A point to which the integral of probability curve is equal to the random number, should be selected.

The SBF at the end of algorithm is named optimality curve. Optimality curve of variable j at point x shows the best achievable fitness for the problem, if *x* be selected for the j^{th} variable.

3.3 Graphical description of the algorithm

In this section, steps of the algorithm are demonstrated by some figures. Figs. 4 to 7 demonstrate various curves of the algorithm for optimizing Allufi-Pentiny function at 1st, 3rd,7th and 25th loop, respectively. Definition of the Allufi-Pentiny function is demonstrated in Table 1.

A typical selection for the steps of the algorithm is as follows:

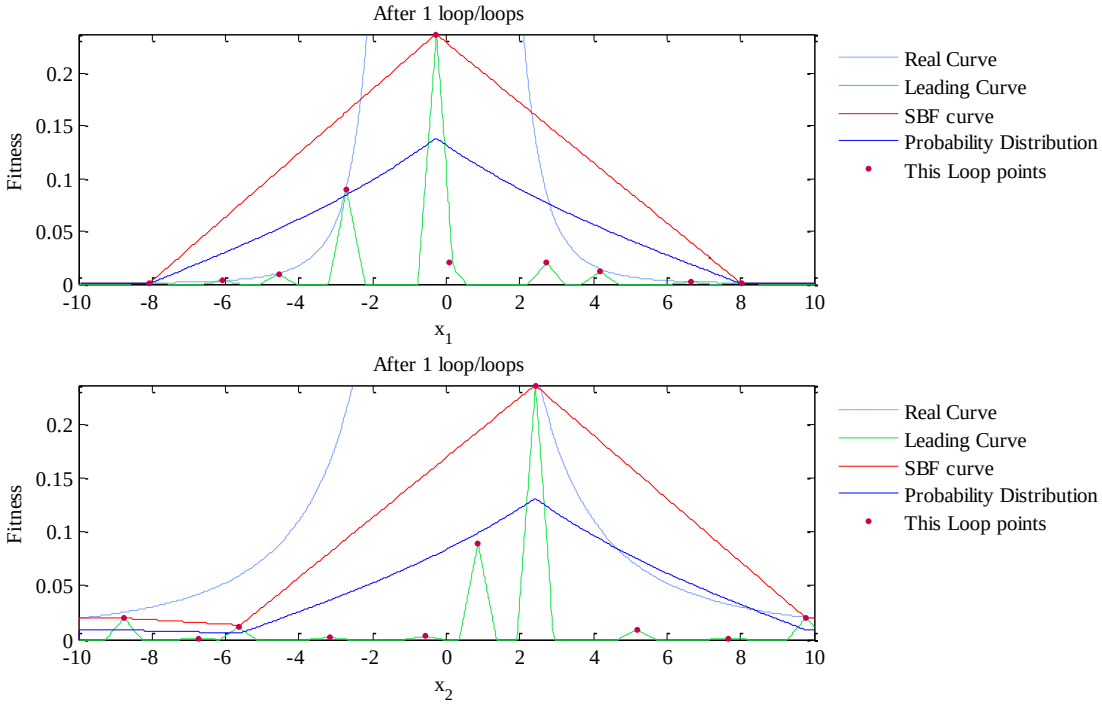

Fig. 4 Graphical demonstration of the DEO curves of Allufi-Pentiny function in the 1st loop of optimization

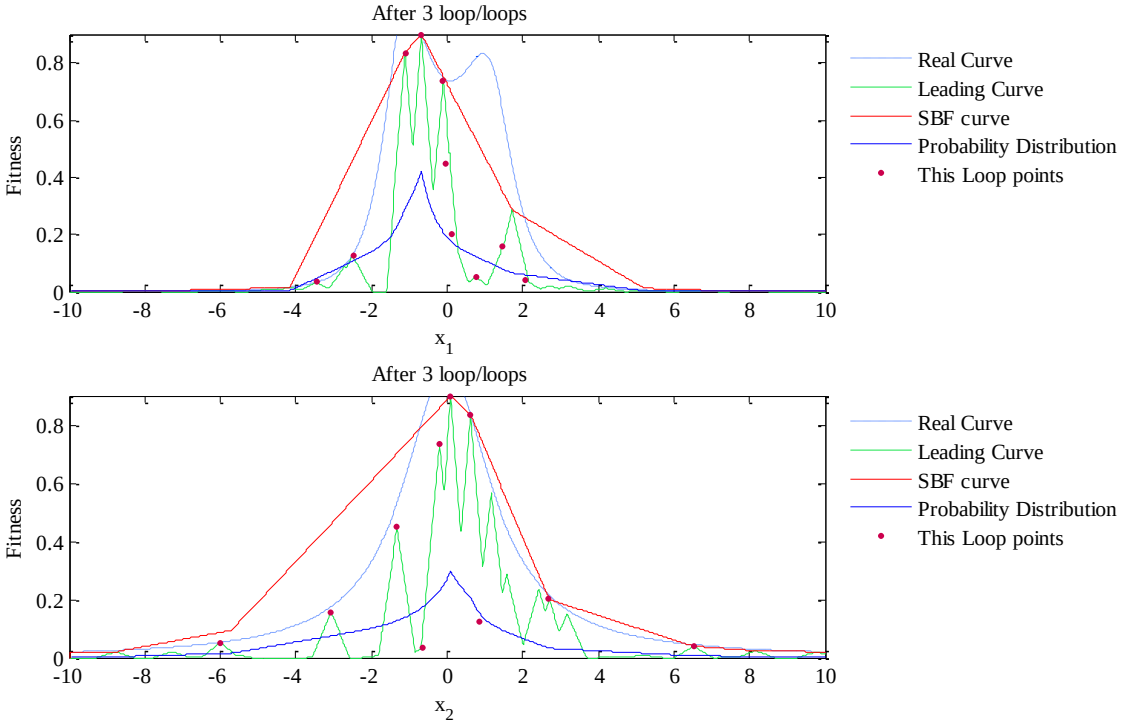

Fig. 5 Graphical demonstration of the DEO curves of Allufi-Pentiny function in the 3rd loop of optimization

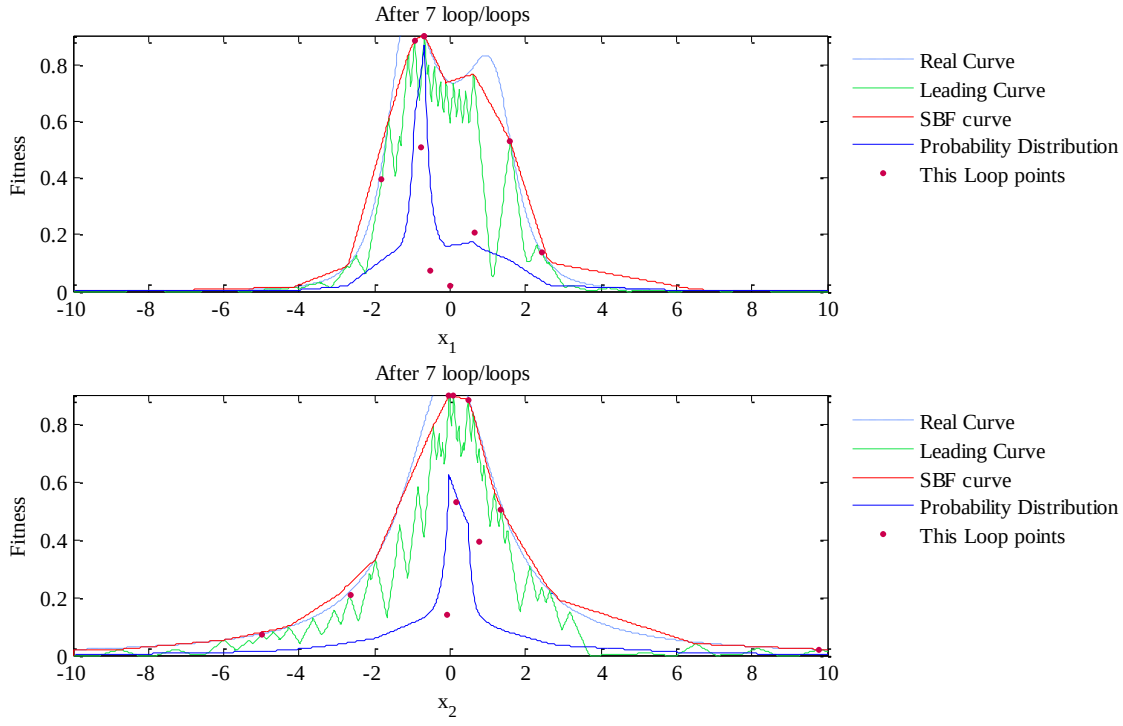

Fig. 6 Graphical demonstration of the DEO curves of Allufi-Pentiny function in the 7th loop of optimization

Table 1 Description of the mathematical benchmarks

Function name	Interval	Function	Global minimum
Aluffi-Pentiny	$X \in [-10,10]^2$	$f(X)=\frac{1}{4}x_1^4-\frac{1}{2}x_1^2+\frac{1}{10}x_1+\frac{1}{2}x_2^2$	-0.352386
Bohachevsky 1	$X \in [-100,100]^2$	$f(X)=x_1^2+2x_2^2-\frac{3}{10}\cos(3\pi x_1)-\frac{4}{10}\cos(4\pi x_2)+\frac{7}{10}$	0.0
Bohachevsky 2	$X \in [-50,50]^2$	$f(X)=x_1^2+2x_2^2-\frac{3}{10}\cos(3\pi x_1).\cos(4\pi x_2)+\frac{3}{10}$	0.0
Becher and Lago	$X \in [-10,10]^2$	$f(X)=(\lvert x_1\rvert-5)^2+(\lvert x_2\rvert-5)^2$	0.0
Branin	$0 \le x_2 \le 15, -5 \le x_1 \le 10$	$f(X)=\left(x_2-\frac{5.1}{4\pi^2}x_1^2+\frac{5}{\pi}x_1\right)^2+10.\left(1-\frac{1}{8\pi}\right)\cos(x_1)+10$	0.397887
Camel	$X \in [-5,5]^2$	$f(X)=4x_1^2-2.1x_1^4+\frac{1}{3}x_1^6+x_1x_2-4x_2^2+4x_2^4$	-1.0316
Cb3	$X \in [-5,5]^2$	$f(X)=2x_1^2-1.05x_1^4+\frac{1}{6}x_1^6+x_1x_2+x_2^2$	0.0
Cosine mixture	$X \in [-1,1]^n, n=4$	$f(X)=\sum_{i=1}^{n}x_i^2-\frac{1}{10}\sum_{i=1}^{n}\cos(5\pi x_i)$	-0.4
DeJoung	$X \in [-5.12,5.12]^3$	$f(X)=x_1^2+x_2^2+x_3^2$	0.0
Exponential	$X \in [-1,1]^n, n=2,4,8$	$f(X)=-\exp\left(-0.5.\sum_{i=1}^{n}x_i^2\right)$	-1.0
Goldstein and price	$X \in [-2,2]^2$	$f(X)=\left[1+(x_1+x_2+1)^2.(19-14x_1+3x_1^2-14x_2+6x_1x_2+3x_2^2)\right]\times \left[30+(2x_1-3x_2)^2.(18-32x_1+12x_1+48x_2-36x_1x_2+27x_2^2)\right]$	3.0
Griewank	$X \in [-100,100]^2$	$f(X)=1+\frac{1}{200}\sum_{i=1}^{2}x_i^2-\prod_{i=1}^{2}\cos\left(\frac{x_i}{\sqrt{i}}\right)$	0.0
Hartman 3	$X \in [0,1]^3$	$f(X)=-\sum_{i=1}^{4}c_i\exp\left(-\sum_{j=1}^{3}a_{ij}(x_j-p_{ij})^2\right)$, $a=\begin{bmatrix}3&10&30\\0.1&10&35\\3&10&30\\0.1&10&35\end{bmatrix}$, $c=\begin{bmatrix}1\\1.2\\3\\3.2\end{bmatrix}$; $p=\begin{bmatrix}0.3689&0.117&0.2673\\0.4699&0.4387&0.747\\0.1091&0.8732&0.5547\\0.03815&0.5743&0.8828\end{bmatrix}$	-3.862782
Rastrigin	$X \in [-1,1]^2$	$f(X)=\sum_{i=1}^{2}\left(x_i^2-\cos(18x_i)\right)$	**-2.0**

1. First, NL points should be selected randomly.

2. Predefined possibility curve should be selected in this step. A linear change in convergence factor is considered and Eq. (2) is used for predefined probability. In this equation, the first loop convergence factor should be calculated and used as PP1, but a value of 10% is assumed for simplicity. If the algorithm be adjusted to finish its works in the 30th loop (Loops Number=30), the predefined probability will be as follows

$$PP(Loop_i)=0.1+0.9*\frac{Loop_i-1}{30-1} \tag{5}$$

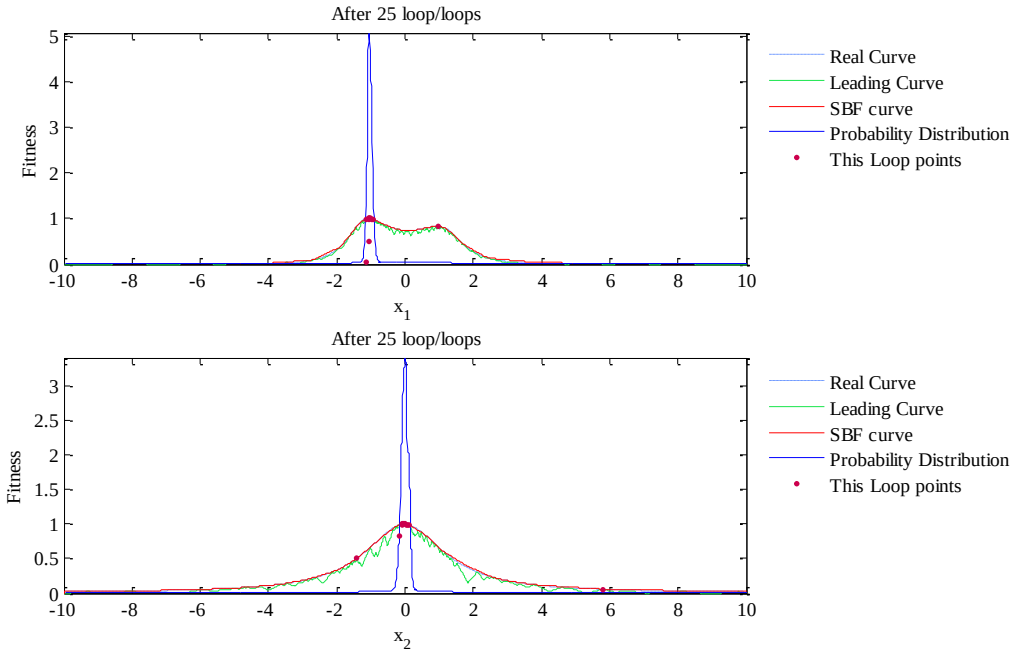

Fig. 7 Graphical demonstration of the DEO curves of Allufi-Pentiny function in the 25th loop of optimization

3. Calculate the fitness of the selected points and locate all red points on both charts as depicted in Fig. 4 (In this example, two variables of x_1 and x_2 are optimized and for each variable there is a chart). It is important to note that each chart is drawn independently.

4. The main goal of this step is to draw a smooth curve which passes through red points. This task contains three steps:

4.1. Create best fitness matrix (BF) and draw Leading curve (LC).

BF is a matrix including all generated locations and their associated fitness. If a location occurs more than once during the optimization process, substitute the fitness value with the greatest one.

Leading curve is created by passing an inverse V-type curve on each of red points. These curves are colored green in Figs. 4 to 7. Base of all these curves are equal to 2*Re. By utilizing these inverse V-types all ignorable points are omitted. In each step, V-type curves of only newly added points should be added to the existing LC.

4.2. Draw Smooth Best Fitness (SBF).

In this step, a smooth curve is constructed over peaks of LC. For having a smooth curve, some points should be ignored and some should be utilized. For performing this task, PoNum parameter is used to select number of ignorable points. In this step for every PoNum peak points (peak points are BF points located on LC curve) a line that is located over all these points is selected. Red curves of Figs. 4 to 7 demonstrate SBF curves.

In the next four steps (5, 6, 7 and 8), the main goal is to form probability curve based on SBF. Here, it should be taken into consideration that probability curve should have a convergence factor according to the pre-defined probability, then SBF should be changed in order to have CF equal to PP. In other words, in each step concentration on better achieved answers should increase.

Graphically it will be possible if one gets peak points and raise the curve in order to increase their probability and consequently decrease dispersion. In final step some random values will be added to the curve for preventing local search and providing a global one.

5. At this step SBF curve should be normalized in order to have maximum value equal to unity and the minimum one equal to zero. It means, the red curve should be sketched in a way that its peak be located at 1 and its minimum be located at 0.

6. All points of curve at this step should be raised to a power greater than unity. After performing this task, the best point of the curve which is equal to 1 in the normalized curve, will remain 1 while the others will decrease by raising to the power. Then the curve will be compressed on the best points.

7. For making the curve of the previous step suitable to be used as a probability curve, total area between the curve and x axis should be equal to 1, then another normalization should be performed by dividing all points value to the total area between the curve and *x* axis. At the end of this step, the total integral of the curve will be equal to 1. This curve is named the Normalized Powered Best Fitness (NPBF) curve.

8. Probability distribution curve is the CF*NPBF+(1−CF)*Random distribution curve. This curve is colored blue in Figs. 4 to 7.

9. Select locations of the next loop according to the probability distribution curve. This means to locate some points randomly in the area between blue curve and *x* axis.

3.4 Input parameters

Input parameters for the algorithm are:

a) Loops number

For an optimization algorithm it is beneficial for the user to be able to enforce the algorithm to work according to the affordable computational cost. The number of loops can be selected by sensitivity analysis when high accuracy is required.

b) Convergence curve formula

This is another important parameter to be selected for the present algorithm. The curve should reach to the final point of 100% smoothly. If the curve satisfies the above mentioned criteria, the algorithm will perform the job properly, but it is recommended to start with a linear curve and try the curves that spend more time (more loops) in high values of *PP*. For example, if one is utilizing proposed curves of this paper, it is recommended to start with *Power*=1 which usually gives good results and it is better to try some cases of the *Power*<1 to check if it improves the results.

c) Effective Radius (R_e)

This parameter should be chosen according to the size of search space and the sensivity of the fitness to each variable.

d) Number of Locations (*NL*)

This parameter is the same as the population size in GA or number of ants in ACO. It should be chosen in a reasonable way.

f) Point numbers (PoNum)

This parameter is used for constructing SBF, which is a smooth curve passing through peaks of LC curve. PoNum helps SBF to ignore some points and do not go up and down with every changes in LC. For example, when PoNum is equal to 5, the algorithm selects 5 of peaks of LC curve and checks how to draw a line which starts from first point and ends to one of points in a way that all other points are located below the curve. In this way, for every 5 points, a line will be

substituted all ups and downs in LC curve. Selection of this parameter does not have significant importance in optimization, but user should avoid large values which decreases accuracy of the curve and final result. Usually it would be nice to set this parameter to 5, but user can increase it by an increase in Number of Locations. Obviously it cannot be more than Number of Locations.

4. Mathematical benchmark functions

In this section DEO is utilized to optimize some mathematical benchmark functions. The description of these functions is illustrated in Table 1. The Experimental setup for the functions are provided in Table 2 and the results achieved by DEO and some other metaheuristics are provided in Table 2. The numbers in Table 3 indicate the average number of function evaluation from 50 independent runs. The numbers in parenthesis demonstrate the ratio of unsuccessful to successful runs. Each run of the algorithm is successful when the results have predefined accuracy, $\varepsilon = \left| f_{\min} - f_{final} \right| = 10^{-4}$. Table 3 illustrates results of DEO in comparison with GA and some of its variants derived from Tsoulos (2008), CSS from Kaveh and Talatahari (2010) and MCSS from Kaveh *et al.* (2013). It can be seen from Table 3 that DEO reaches to the final point for all functions in 4239 iterations, while CSS and MCSS do the same work in 6943 and 14132 iterations, respectively and GA perform it in more than 20000 iterations.

Optimality curves of some benchmark functions are demonstrated in Figs. 8 to 11. In all these figures the real values of the optimality curves are calculated from exact methods and are drawn as dashed lines. It can be seen that DEO can guess optimality curve with a reasonable accuracy.

Table 2 Experimental setup for the functions

Function	Nloc	LoopNumber
Aluffi-Pentiny	10	20
Bohachevsky 1	20	20
Bohachevsky 2	20	20
Becher and Lago	10	10
Branin	10	20
Camel	10	20
Cb3	10	15
CM	20	20
Dejoung	10	20
Exp2	10	8
Exp4	10	15
Exp8	20	25
GP	20	20
Griewank	20	50
Hartman 3	10	25
Rastrigin	10	30

For all functions, the Power is equal to 1, and the PoNum is taken as 5

A 200-bar planar truss

A 200-bar truss shown in Fig. 12 is another problem to be optimized by DEO. All members are made of steel: the material density and modulus of elasticity are 0.283 lb/in^3 (7933.410 kg/m^3) and 30,000 ksi (206,000 MPa), respectively. The truss is subjected to stress limitations of ±10 ksi (68.95 MPa) only. Three loading condition is considered: (1) 1.0 kip (4.45 kN) acting in the positive *x*-direction at nodes 1, 6, 15, 20, 29, 34, 43, 48, 57, 62 and 71. (2) 10 kips (44.5 kN) acting

Table 3 Performance comparison for the benchmark problems

Function	GEN	GEN-S	GEN-S-M	GEN-S-M-LS	CSS	MCSS	DEO
AP	1,360 (0.99)	1,360	1,277	1,253	804	316	134
Bf1	3,992	3,356	1,640	1,615	1,187	464	315
Bf2	20,234	3,373	1,676	1,636	742	425	328
BL	19,596	2,412	2,439	1,436	423	361	100
Branin	1,442	1,418	1,404	1,257	852	332	182
Camel	1,358	1,358	1,336	1,300	575	342	156
Cb3	9,771	2,045	1,163	1,118	436	267	119
CM	2,105	2,105	1,743	1,539	1,563	421	301
Dejoung	9,900	3,040	1,462	1,281	630	334	160
Exp2	938	936	817	807	132	146	59
Exp4	3,237	3,237	2,054	1,496	867	284	140
Exp8	3,237	3,237	2,054	1,496	1,426	553	460
GP	1,478	1,478	1,408	1,325	682	358	337
Griewank	18838 (0.91)	3,111 (0.91)	1,764	1,652 (0.99)	1,551	976	952
Hartman 3	1,350	1,350	1,332	1,274	860	322	222
Rastrigin	1,533 (0.97)	1,523 (0.97)	1,392	1,381	1,402	1,042	277
Total	112,311 (96.7)	41,640 (96.7)	29,166 (98.16)	25,193 (98.16)	17,367	6,943	4,239

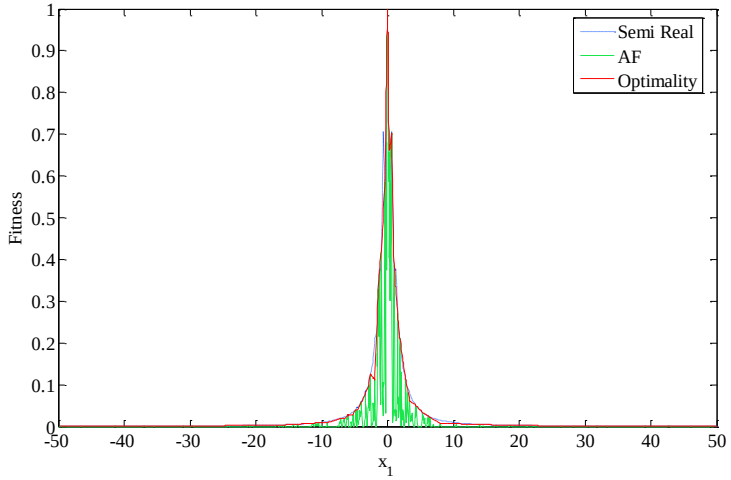

Fig. 8 Bohachevsky 2 function real curve, leading curve and optimality curve for x_1

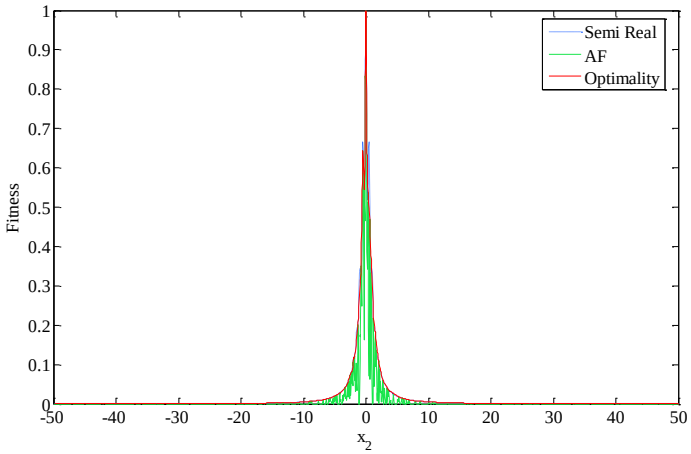

Fig. 9 Bohachevsky 2 function real curve, leading curve and optimality curve for x_2

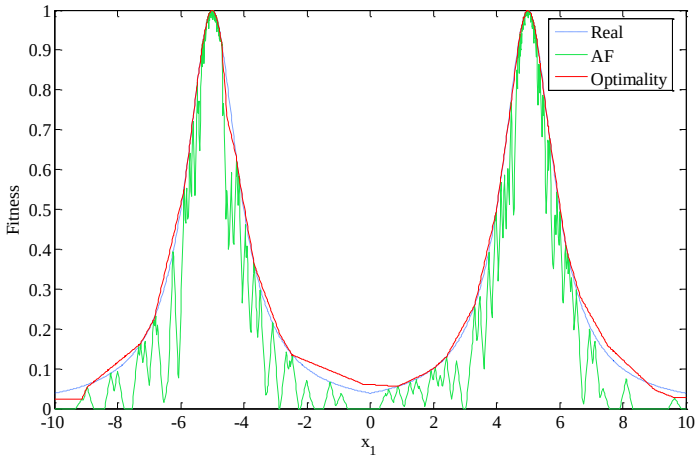

Fig. 10 Becker and Lago function real curve, leading curve and optimality curve for x_1

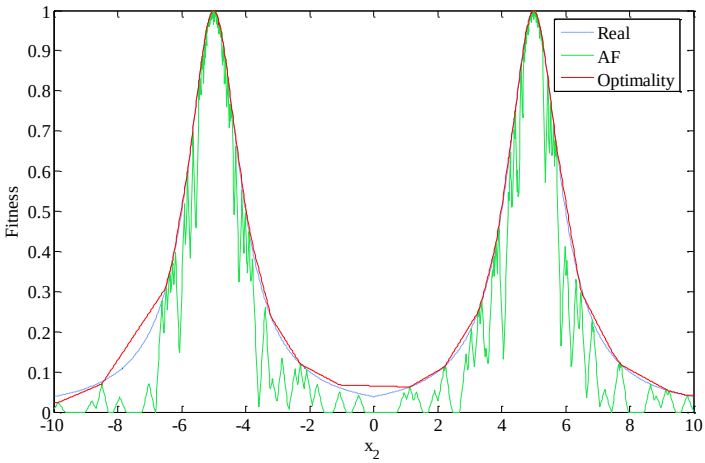

Fig. 11 Becker and Lago function Semi real, real curve, leading curve and optimality curve for x_2

in the negative *y*-direction at nodes 1, 2, 3, 4, 5, 6, 8, 10, 12, 14, 15, 16, 17, 18, 19, 20, 22, 24, …, 71, 72, 73, 74 and 75; and (3) Conditions (1) and (2) acting together. The 200 members of this truss are divided into 29 groups, as shown in Table 4. The minimum cross-sectional area of all members is 0.1 in^2 (0.6452 cm^2) and the maximum cross-sectional area is 20 in^2 (129.03 cm^2).

Table 4 illustrates the results of the DEO in comparison with those of the PSO and some of its variants derived from Kaveh and Talatahari (2009) and Ray optimization from Kaveh and Khayatazad (2013). Results are presented in Table 4. It can be seen from this table that DEO achieves better results compared to PSO, PSOPC, HPACO and Ray optimization algorithm. Optimality curves of 5th, 15th and 29th member groups are demonstrated in Figs. 13 to 15. In these figures it can be seen that optimum answer is on top of a curve and slope of the left side is more than the right one. In left side of optimum point, fitness decreases by an increase in penalties and in the right side, it decreases by an increase in size of members and consequently weight of the structure. In addition changes in fitness during the optimization are depicted in Fig. 16.

Fig. 17 illustrates the optimization process and convergence of DEO in comparison to CSS and MCSS. It can be seen that DEO has higher convergence rate and also it can reach to the global minimum in all mathematical functions. It should be added that in each iteration, the number of function evaluations is equal to number of locations or NL.

Table 4 Optimum design comparison for the 200-bar planar truss

	Group variables members (A_i, i=1,…,200)	Optimal cross -sectional area (in^2)					
		Kaveh and Talatahari (2010)			Kaveh and Khayatazad (2013)	Present work	
		PSO	PSOPC	HPACO	RO	(in^2)	(cm^2)
1	1, 2, 3, 4	0.8016	0.7590	0.1033	0.3820	0.1046	0.6750
2	5, 8, 11, 14, 17	2.4028	0.9032	0.9184	2.1160	0.9058	5.8439
3	19, 20, 21, 22, 23, 24	4.3407	1.1000	0.1202	0.1020	0.1285	0.8290
4	18, 25, 56, 63, 94, 101, 132, 139, 170, 177	5.6972	0.9952	0.1009	0.1410	0.1114	0.7190
5	26, 29, 32, 35, 38	3.9538	2.1350	1.8664	3.6620	1.8615	12.0099
6	6, 7, 9, 10, 12, 13, 15, 16, 27, 28, 30, 31, 33, 34, 36, 37	0.5950	0.4193	0.2826	0.1760	0.2826	1.8234
7	39, 40, 41, 42	5.6080	1.0041	0.1000	0.1210	0.1065	0.6870
8	43, 46, 49, 52, 55	9.1953	2.8052	2.9683	3.5440	2.9609	19.1024
9	57, 58, 59, 60, 61, 62	4.5128	1.0344	0.1000	0.1080	0.1052	0.6790
10	64, 67, 70, 73, 76	4.6012	3.7842	3.9456	5.5650	3.9570	25.5287
11	44, 45, 47, 48, 50, 51, 53, 54, 65, 66, 68, 69, 71, 72, 74, 75	0.5552	0.5269	0.3742	0.5420	0.3701	2.3880
12	77, 78, 79, 80	18.7510	0.4302	0.4501	0.1380	0.4517	2.9145
13	81, 84, 87, 90, 93	5.9937	5.2683	4.9603	5.1390	4.9618	32.0117
14	95, 96, 97, 98, 99, 100	0.1000	0.9685	1.0738	0.1010	1.0858	7.0049
15	102, 105, 108, 111, 114	8.1561	6.0473	5.9785	8.7420	5.9672	38.4979
16	82, 83, 85, 86, 88, 89, 91, 92, 103, 104, 106, 107, 109, 110, 112, 113	0.2712	0.7825	0.7863	0.4310	0.7708	4.9730
17	115, 116, 117, 118	11.1520	0.5920	0.7374	0.9980	0.7231	4.6649
18	119, 122, 125, 128, 131	7.1263	8.1858	7.3809	7.7120	7.3825	47.6287
19	133, 134, 135, 136, 137, 138	4.4650	1.0362	0.6674	0.1520	0.6620	4.2710
20	140, 143, 146, 149, 152	9.1643	9.2062	8.3000	8.4520	8.3085	53.6030
21	120, 121, 123, 124, 126, 127, 129, 130, 141, 142, 144, 145, 147, 148, 150, 151	2.7617	1.4774	1.1967	0.8350	1.1950	7.7097
22	153, 154, 155, 156	0.5541	1.8336	1.0000	0.4130	1.0083	6.5049
23	157, 160, 163, 166, 169	16.1640	10.6110	10.8262	10.1460	10.8226	69.8234
24	171,172,173,174,175,000	0.4974	0.9851	0.1000	0.8740	0.1108	0.7150
25	178, 181, 184, 187, 190	16.2250	12.5090	11.6976	11.3840	11.6834	75.3769
26	158, 159, 161,162 , 164, 165, 167, 168, 179, 180, 182, 183, 185, 186, 188, 189	1.0042	1.9755	1.3880	1.1970	1.3734	8.8607
27	191, 192, 193, 194	3.6098	4.5149	4.9523	5.7470	4.9415	31.8804
28	195, 197, 198,200	8.3684	9.8000	8.8000	7.8230	8.7942	56.7366
29	196, 199	15.5620	14.5310	14.6645	13.6550	14.6726	94.6616
	Weight (lb)	44081.40	28537.80	25156.50	25193.22	25125.54	111688.30

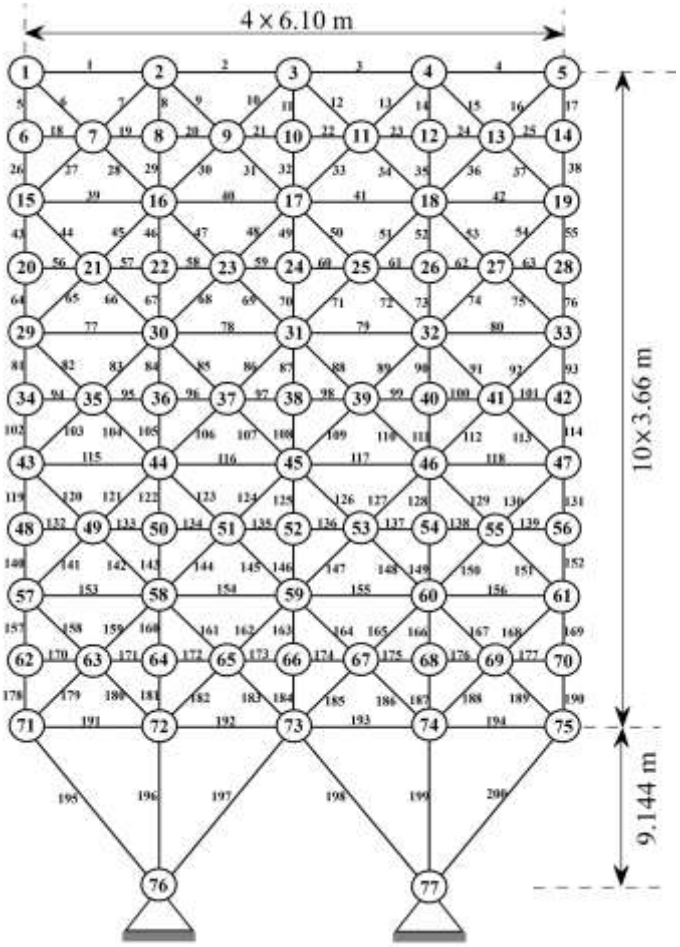

Fig. 12 The schematic of a 200-bar planar truss

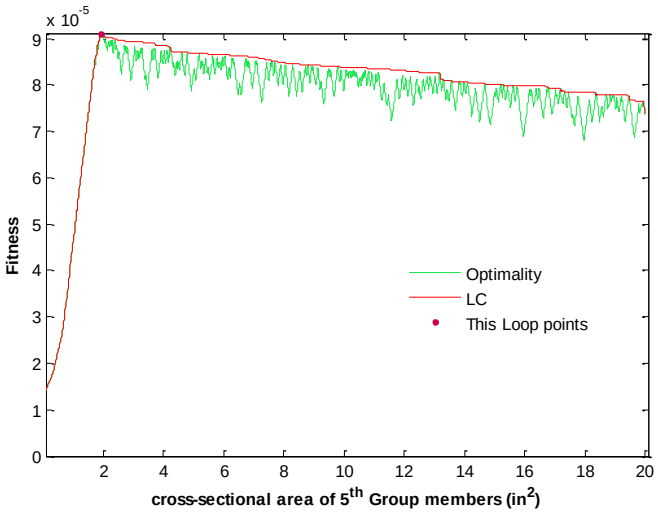

Fig. 13 Optimality curve of the 5th group members

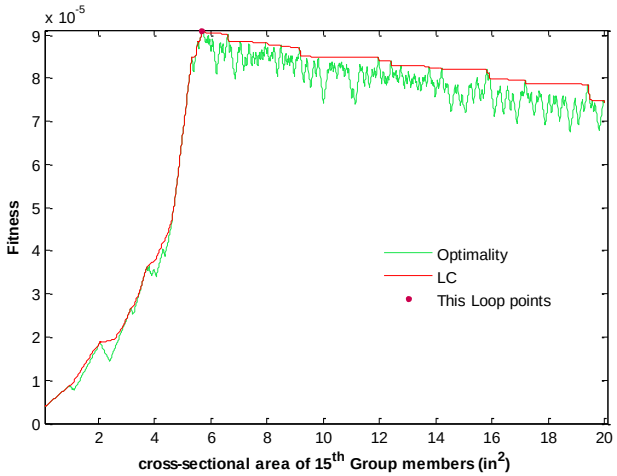

Fig. 14 Optimality curve of the 15th group members

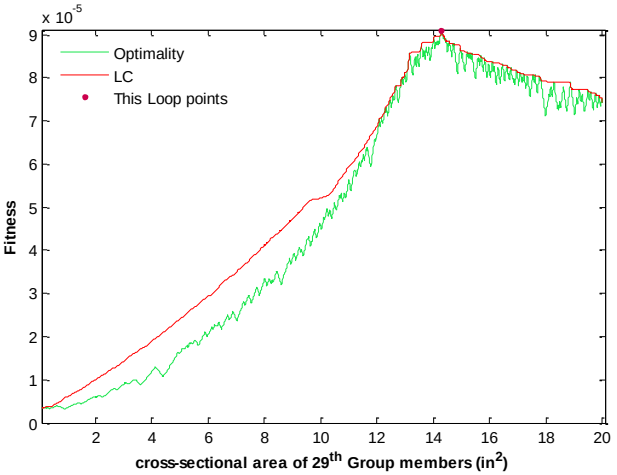

Fig. 15 Optimality curve of the 29th group members

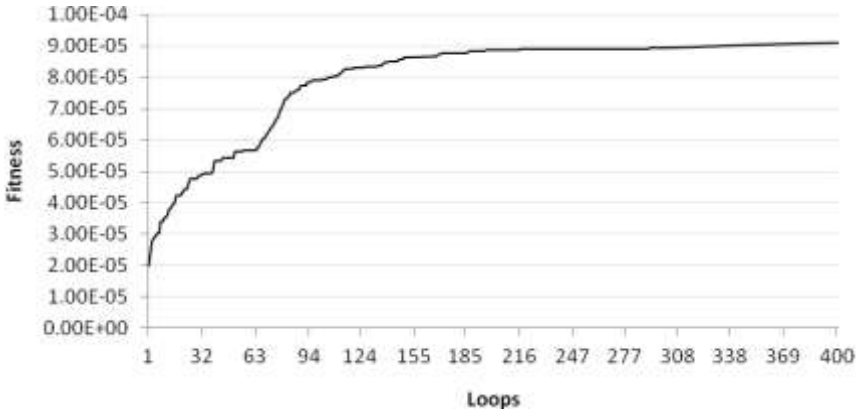

Fig. 16 Chnages in the fitness during the optimization of 200-bar truss

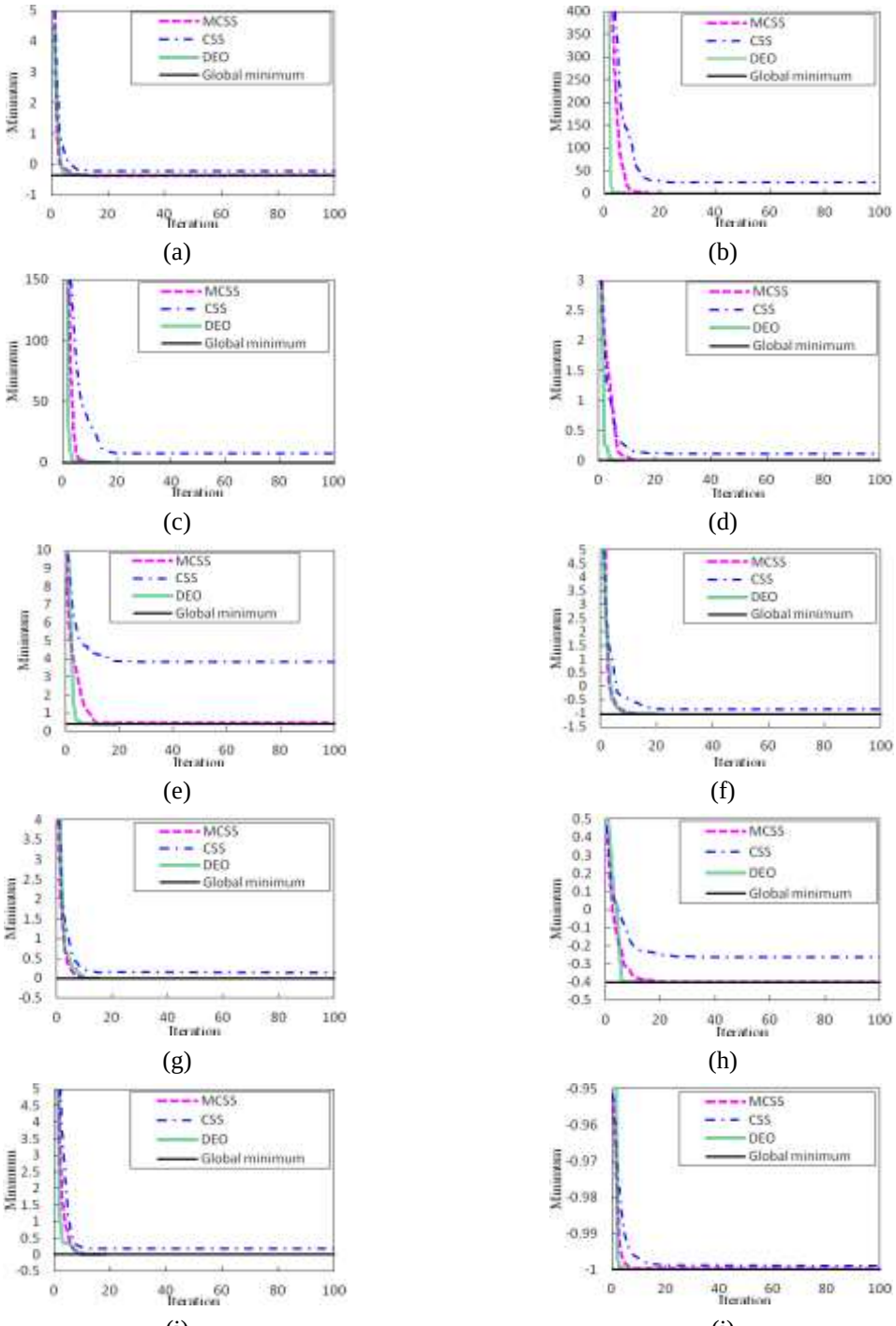

Fig. 17 Comparison of the convergance rate of optimizing mathematical benchmarks; (a) AP, (b) Bf1, (c) Bf2, (d) BL, (e) Branin, (f) Camel, (g) Cb3, (h) CM, (i) Dejoung, (j) Exp2, (k) Exp4, (l) Exp8, (m) Goldstein and price, (n) Griewank, (o) Hartman3, (p) Rastrigin

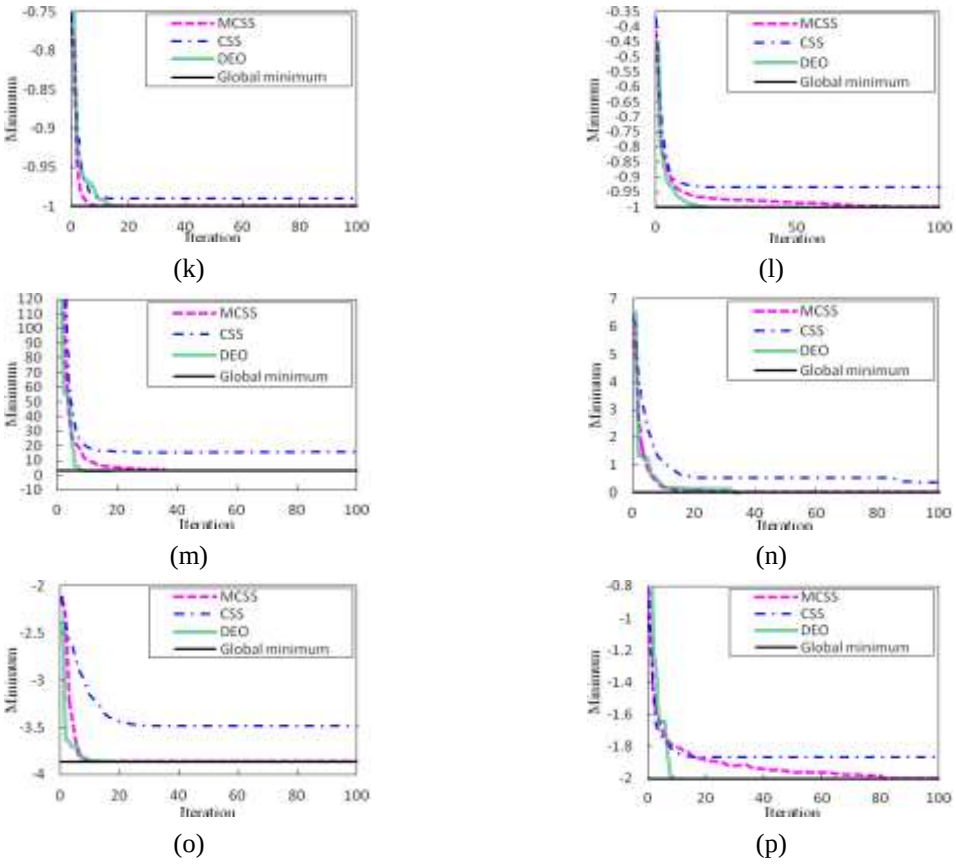

Fig. 17 Continued

5. Conclusions

In this study, Dolphin Echolocation Optimization algorithm (DEO) is presented for optimization in continuous search space. The most important feature of present algorithm is its ability in developing optimality curves which can be used as a guide for designers. In these curves, not only the optimum choice for each variable is depicted but also change in the fitness function due to changes in each variable is demonstrated.

Another ability of the presented method is that it is adjustable for a pre-determined computational cost. It is also self adaptive and it has only a few parameters to be set. Perhaps the most important feature of the present algorithm is its ability in obtaining optimality curves.

In this study, DEO is utilized for optimization of mathematical and engineering problems. Results show that DEO often achieves better results with higher convergence rates compared to some existing meta-heuristic algorithms previously applied to these problems.

References

Au, W.W.L. and Simmons, J. (2007), "Echolocation in dolphins and bats", *Phys Today*, **60**(9), 40-45.

Dorigo, M, Maniezzo, V. and Colorni, A. (1996), "The ant system: optimization by a colony of cooperating agents", *IEEE Trans. Syst. Man Cybern.*, B**26**, 29-41.
Eberhart, R.C. and Kennedy, J. (1995), "A new optimizer using particle swarm theory", *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, Nagoya, Japan.
Erol, O.K. and Eksin, I. (2006), "New optimization method: Big Bang-Big Crunch", *Adv. Eng. Softw.*, **37**(2), 106-111.
Fogel, L.J., Owens, A.J. and Walsh, M.J. (1966), *Artificial intelligence through simulated evolution*, Wiley, Chichester, UK.
Goldberg, D.E. (1989), *Genetic algorithms in search optimization and machine learning*, Addison-Wesley, Boston, USA.
Gonçalves, M.S., Lopez, R.H. and Miguel, L.F.F. (2015), "Search group algorithm: A new metaheuristic method for the optimization of truss structures", *Comput. Struct.*, **153**, 165-184.
Holland, J.H. (1975), *Adaptation in natural and artificial systems*, University of Michigan Press, Ann Arbor.
Lee, K.S. and Geem, Z.W. (2004) "A new structural optimization method based on the harmony search algorithm", *Comput. Struct.*, **82**(9), 781-798.
Kaveh, A. and Farhoudi, N. (2013), "A new optimization method: Dolphin echolocation", *Adv. Eng. Softw.*, **59**, 53-70.
Kaveh, A. and Farhoudi, N. (2011), "A unified approach to parameter selection in meta-heuristic algorithms for layout optimization", *J. Constr. Steel Res.*, **67**(10), 15453-15462.
Kaveh, A. and Ilchi Ghazaan, M. (2015), "Truss optimization with dynamic constraints using UECBO", *Adv. Comput. Des., Techno Press*, Accepted for publication, 2015.
Kaveh, A. and Khayatazad, M. (2013), "Ray optimization for size and shape optimization of truss structures", *Comput. Struct.*, **117**, 82-94.
Kaveh, A. and Mahdavai, V.R. (2014), "Colliding bodies optimization method for optimum design of truss structures with continuous variables", *Adv. Eng. Softw.*, **70**, 1-12.
Kaveh, A. and Maniat, M. (2015), "Damage detection based on MCSS and PSO using modal data", *Smart Struct. Syst.*, **15**(5), 1253-1270.
Kaveh, A. and Talatahari, S. (2009), "Particle swarm optimizer, ant colony strategy and harmony search scheme hybridized for optimization of truss structures", *Comput. Struct.*, **87**(5), 267-283.
Kaveh, A. and Talatahari, S. (2010), "A novel heuristic optimization method: charged system search", *Acta Mech.*, **213**(3-4), 267-286.
Kaveh, A. and Zolghadr, A. (2014), "A new PSRO algorithm for frequency constraint truss shape and size optimization", *Struct. Eng. Mech.*, **52**(3), 445-468.
Koza, J.R. (1990), *Genetic programming: a paradigm for genetically breeding populations of computer programs to solve problems*, Report No. STAN-CS-90-1314, Stanford University, Stanford, CA.
May, J. (1990), *The greenpeace book of dolphins*, Greenpeace Communications Ltd.
Mirjalili, S. (2015), "The ant lion optimizer", *Adv. Eng. Softw.*, **83**, 80-98.
Rao, R.V., Savsani, V.J. and Vakharia, D.P. (2011) "Teaching-learning-based optimization: A novel method for constrained mechanical design optimization problems", *Comput. Aid. Des.*, **43**(3), 303-315.
Sadollah, A., Eskandar, H., Bahreininejad, A. and Kim, J.H. (2015), "Water cycle, mine blast and improved mine blast algorithms for discrete sizing optimization of truss structures", *Comput. Struct.*, **149**, 1-16.
Tsoulos, I.G. (2008), "Modifications of real code genetic algorithm for global optimization", *Appl. Math. Comput.*, **203**(2), 598-607.
Yang, X.S. (2010), "A new metaheuristic bat-inspired algorithm", *NICSO*, 284, 65-74.
Yang, X.S. and Deb, S. (2009), "Engineering optimisation by cuckoo search", *Int. J. Math. Model. Num. Optim.*, **1**(4), 330-343.
Yang, X.S. (2011), "Bat algorithm for multi-objective optimization", *Int. J. Bio-Inspired Comput.*, **3**(5), 267-274.